2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)
July 6-11, 2014, Beijing, China

Hybrid Fuzzy Genetics-based Machine Learning with Entropy-based Inhomogeneous Interval Discretization

Yuji Takahashi, Yusuke Nojima, and Hisao Ishibuchi

***Abstract*- Discretization of continuous attributes is a key issue in classifier design from numerical data. In the machine learning community, continuous attributes are discretized into intervals. An entropy measure is often used to determine the cutting points for interval discretization. In the fuzzy system community, continuous attributes are usually discretized into overlapping fuzzy sets. Learning and optimization techniques are used to adjust the membership function of each fuzzy set. One interesting research issue is a comparison between interval partitions and fuzzy partitions. We address this issue by using an entropy-based interval discretization method in hybrid fuzzy genetics-based machine learning (GBML). Our hybrid fuzzy GBML algorithm is applied to a number of data sets where interval discretization is fuzzified with different fuzzification grades from zero (i.e., interval partitions) to one (i.e., completely fuzzified partitions). Experimental results from various fuzzification grades are compared with each other.**

I. Introduction

Applications of genetic algorithms to machine learning are often referred to as genetics-based machine learning (GBML [1]). When classification problems with continuous attributes are handled by GBML, each attribute is divided into intervals. This process is called interval discretization, which is also used in many models of machine learning such as decision trees [2]-[4]. Although interval discretization looks a nice idea to handle continuous attributes in classifier design, it is not always appropriate for any data sets. For example, some attributes are noisy or vague. In such a case, multiple pattern distributions with different classes are overlapping one other. To handle this situation, each attribute is often divided into overlapping fuzzy sets. Those fuzzy sets are used in the antecedent (i.e., condition) part of a fuzzy if-then rule to specify a fuzzy region in the pattern space.

The use of GBML for fuzzy rule-based classifier design has been actively studied under the name of evolutionary fuzzy systems or genetic fuzzy systems [5], [6]. Fuzzy rules are often categorized into two types: linguistic fuzzy rules and approximate fuzzy rules. Linguistic fuzzy rules are based on the use of pre-specified linguistic terms such as *small*, *medium* and *large* as antecedent fuzzy sets. Thus each rule is highly interpretable. Approximate fuzzy rules do not use those pre-specified linguistic terms, but these rules use adjustable fuzzy sets to pattern distributions.

In our hybrid fuzzy GBML algorithm [7], [8], we always used homogeneous fuzzy partitions in Fig. 1 to design fuzzy rule-based classifiers with high interpretability. In this paper, we examine the effect of inhomogeneous fuzzy partitions in our hybrid fuzzy GBML algorithm. An inhomogeneous fuzzy partition is derived from each interval partition obtained by an entropy-based interval discretization scheme. Examples of inhomogeneous fuzzy partitions are shown in Fig. 2 where an interval discretization of each attribute is shown by vertical dashed lines. In [9], [10], the effect of inhomogeneous fuzzy partitions derived from interval partitions was examined in genetic fuzzy rule selection so far. It was shown that the fuzzification of intervals improved the generalization ability of interval rules while it slightly deteriorated the accuracy on training patterns [10].

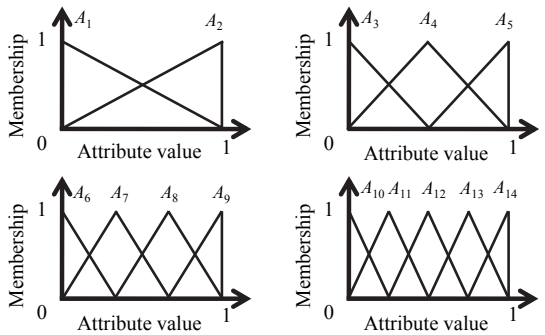


Fig. 1. Homogeneous fuzzy partitions with different granularity.

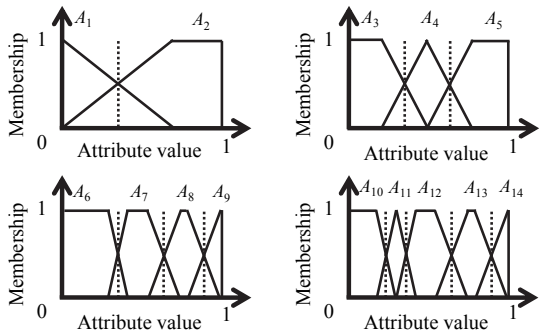


Fig. 2. Inhomogeneous fuzzy partitions with different granularity which are derived from the interval partitions shown by vertical dashed lines.

In this paper, we use inhomogeneous fuzzy partitions in our hybrid fuzzy GBML algorithm to examine the effect of fuzzification of intervals on the performance of rule-based classifiers. We also compare inhomogeneous fuzzy partitions with homogeneous ones.

Yuji Takahashi is with Department of Computer Science and Intelligent Systems, Osaka Prefecture University, Japan (phone: +81-72-254-9198; fax: +81-72-254-9915; e-mail: yuji.takahashi@ci.cs.osakafu-u.ac.jp).

Yusuke Nojima and Hisao Ishibuchi are with Department of Computer Science and Intelligent Systems, Osaka Prefecture University, Japan (e-mail: {nojima, hisaoi}@cs.osakafu-u.ac.jp).

This work was partially supported by Grand-in-Aid for Scientific Research (C): KAKENHI (25330292).

978-1-4799-2072-3/14/$31.00 ©2014 IEEE

This paper is organized as follows. First, we explain fuzzy rule-based classifiers in Section II. Next we explain how to discretize continuous attributes into intervals using an entropy measure in Section III where we also explain the fuzzification of discretized intervals. In Section IV, we explain GBML in general and our hybrid fuzzy GBML algorithm in particular. Then we apply our algorithm to a number of data sets using interval and fuzzy partitions in Section V. Finally, we conclude this paper in Section VI.

II. Fuzzy Rule-based Classifiers

In this section, we explain a fuzzy rule-based classifier. Let us assume that we design a classifier for an M-class data set with m training patterns $\mathbf{x}_p = (x_{p1}, \ldots, x_{pn})$, $p = 1, 2, \ldots, m$. Each pattern has n attribute values and a class label. For simplicity, each attribute value x_{pi} is normalized into a real number in [0, 1] for $i = 1, 2, ..., n$ and $p = 1, 2, \ldots, m$. Thus, all the training patterns are located in an n-dimensional pattern space $[0, 1]^n$. For this classification problem, we use if-then rules of the following type [11]:

$$\text{Rule } R_q\text{: If } x_1 \text{ is } A_{q1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{qn} \text{ then Class } C_q \text{ with } CF_q, \quad (1)$$

where R_q is a rule label for the qth rule, A_{qi} is an antecedent fuzzy set or interval set, C_q is a class label, and CF_q is a rule weight. As an antecedent set, we simultaneously use four partitions with different granularity and "*don't care*". This is because the appropriate number of partitions always depends on the distribution of patterns in each attribute. Fig. 1 shows four kinds of homogeneous fuzzy partitions which have 14 fuzzy sets in total. Fig. 2 shows four kinds of inhomogeneous fuzzy partitions derived from interval partitions generated by using an entropy-based discretization. We will explain how to make these partitions later in the next section. "*don't care*" is always fully compatible with any attribute values.

The consequent class C_q and the rule weight CF_q of each rule R_q can be specified in a heuristic manner using compatible training patterns with its antecedent part [12]-[14]. First, we calculate the confidence of the rule "$\mathbf{A}_q \Rightarrow$ Class h" for each class h as follows:

$$c(\mathbf{A}_q \Rightarrow \text{Class } h) = \frac{\sum_{\mathbf{x}_p \in \text{Class } h} \mu_{\mathbf{A}_q}(\mathbf{x}_p)}{\sum_{p=1}^{m} \mu_{\mathbf{A}_q}(\mathbf{x}_p)}, \quad h = 1, 2, ..., M, \quad (2)$$

where $\mu_{\mathbf{A}_q}(\mathbf{x}_p)$ is the compatibility grade of the rule q, which is calculated as

$$\mu_{\mathbf{A}_q}(\mathbf{x}_p) = \mu_{A_{q1}}(x_{p1}) \cdot \mu_{A_{q2}}(x_{p2}) \cdot \ldots \cdot \mu_{A_{qn}}(x_{pn}). \quad (3)$$

The consequent class C_q is specified as the class which has the maximum confidence:

$$c(\mathbf{A}_q \Rightarrow \text{Class } C_q) = \max_{h=1, 2, ..., M} c(\mathbf{A}_q \Rightarrow \text{Class } h). \quad (4)$$

By using the confidence measure, the rule weight CF_q is calculated as follows:

$$CF_q = c(\mathbf{A}_q \Rightarrow \text{Class } C_q) - \sum_{\substack{h=1 \\ h \neq C_q}}^{M} c(\mathbf{A}_q \Rightarrow \text{Class } h). \quad (5)$$

A classifier has a number of if-then rules explained above. Let S be a classifier (i.e., a set of if-then rules). When an input pattern $\mathbf{x}_p$ is to be classified by S, a single winner rule R_{win} is chosen from S using the product of the compatibility grade in (3) and the rule weight of each if-then rule in S as follows:

$$\mu_{\mathbf{A}_{\text{win}}}(\mathbf{x}_p) \cdot CF_{\text{win}} = \max\{\eta_q(\mathbf{x}_p) \mid R_q \in S\}, \quad (6)$$

$$\eta_q(\mathbf{x}_p) = \mu_{\mathbf{A}_q}(\mathbf{x}_p) \cdot CF_q. \quad (7)$$

The input pattern $\mathbf{x}_p$ is assigned the consequent class C_{win} of the winner rule R_{win}. When multiple rules with different consequent classes have the same maximum value in (6), the classification of the input pattern $\mathbf{x}_p$ is rejected. When no rules cover $\mathbf{x}_p$, the classification of $\mathbf{x}_p$ is also rejected [15].

III. Entropy-based Discretization

In this section, we explain how to generate inhomogeneous interval partitions and fuzzy ones.

A. Entropy-based Discretization

A number of discretization methods have been proposed in the literature [2], [4], [16]. We use the class entropy measure defined in the following equation to divide a continuous attribute into K intervals [2]:

$$H(A_1, ..., A_K) = -\sum_{j=1}^{K} \frac{|D_j|}{|D|} \sum_{h=1}^{M} \left(\frac{|D_{jh}|}{|D_j|} \cdot \log_2 \frac{|D_{jh}|}{|D_j|} \right), \quad (8)$$

where A_j is an interval, D_j is the set of input training patterns in the interval A_j, and D_{jh} is the set of the training patterns of Class h in D_j. The number of training patterns in each subset is denoted by $|\cdot|$ such as $|D_j|$ in (8). By minimizing this class entropy measure, each interval tends to include patterns with a single class. As we explained in the previous section, we generate four kinds of interval partitions corresponding to K = 2, 3, 4, and 5.

B. Inhomogeneous Fuzzy Partitions Derived from Interval Partitions

In this subsection, we explain a general framework for defining fuzzy partitions from interval partitions. The idea of fuzzy discretization derived from interval discretization was first proposed in [9]. According to the following constraint conditions, we can generate inhomogeneous fuzzy partitions:

(a) Membership functions are linear (i.e., triangular or trapezoidal).
(b) The sum of the membership values of neighboring fuzzy sets is 1.
(c) Crossing points of neighboring membership functions are identical to threshold values for interval.

(d) The membership value of the intermediate fuzzy sets (e.g., MS: *medium small*, M: *medium*, and ML: *medium large* in Fig. 3) is 1 at the midpoint of the corresponding intervals.
(e) The membership value of the smallest fuzzy set (e.g., S: *small* in Fig. 3) is 1 at the smallest input value (i.e., 0 in Fig. 3) in the domain interval.
(f) The membership value of the largest fuzzy set (e.g., L: *large* in Fig. 3) is 1 at the largest input value in the domain interval (i.e., 1 in Fig. 3).

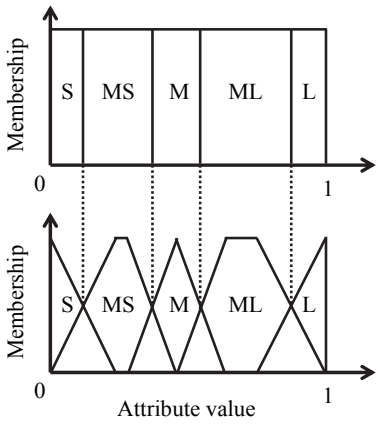


Fig. 3. Interval discretization with F = 0 (the upper part) and fuzzy discretization with F = 1 (the lower part) when K = 5.

Fig. 3 shows an example of fuzzy discretization derived from crisply discretized interval. It should be noted that we cannot uniquely specify the fuzzy discretization by the above six constraint conditions. To define fuzzy discretization uniquely, we have to consider the fuzzification grade F of fuzzy discretization, which means the degree of overlap of the neighboring partitions. When we specify F as 1, fuzzy discretization is completely fuzzified under the above six constraint conditions as shown in Fig. 3. In contrast, $F = 0$ corresponds to interval partition which has no overlaps between adjacent fuzzy sets. From complete fuzzy discretization with $F = 1$ and interval partition with $F = 0$, we can generate optionally fuzzified discretization with arbitrary grades of fuzzification (i.e., fuzzification grade).

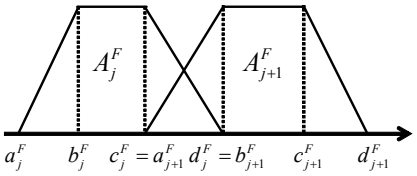


Fig. 4. Adjacent trapezoidal fuzzy sets.

Let us denote optionally fuzzified trapezoidal fuzzy set with fuzzification grade F as $A_j^F = (a_j^F, b_j^F, c_j^F, d_j^F)$ where $0 \leq F \leq 1$ (see Fig. 4). Here, four parameters (i.e., a_j^F, b_j^F, c_j^F, d_j^F) mean four vertices of the trapezoid. If the fuzzy set corresponds to triangle, b_j^F and c_j^F are the same value. Note that A_j^0 and A_j^1 correspond to crisp discretization and completely fuzzified discretization, respectively.

Using the interpolation between A_j^0 and A_j^1, we can uniquely specify the optionally fuzzified discretization $A_j^F = (a_j^F, b_j^F, c_j^F, d_j^F)$ as the following equations:

$$a_j^F = a_j^0 + (a_j^1 - a_j^0) \cdot F, \tag{9}$$

$$b_j^F = b_j^0 + (b_j^1 - b_j^0) \cdot F, \tag{10}$$

$$c_j^F = c_j^0 + (c_j^1 - c_j^0) \cdot F, \tag{11}$$

$$d_j^F = d_j^0 + (d_j^1 - d_j^0) \cdot F. \tag{12}$$

IV. Fuzzy GBML Algorithms

A. GBML Algorithms

In this subsection, we explain GBML algorithms. GBML algorithms can be divided into two categories depending on their coding mechanisms [17]. One is the Pittsburgh approach where a rule set (i.e., a classifier) is coded as a string and handled as an individual. A population consists of a number of rule sets. Pittsburgh-style GBML algorithms search for the best rule set with respect to a pre-specified fitness function. Thus, classifiers are directly optimized. The other category is the Michigan approach where an individual represents a rule, while the population is a rule set. Since a number of rules are optimized, a classifier includes well-designed rules.

As the combination of the above two types of GBML algorithms, we proposed the hybridization of Michigan and Pittsburgh-style GBML algorithms [8]. In this study, we use this hybrid GBML algorithm.

B. Hybrid Fuzzy GBML Algorithm

In this subsection, we explain our hybrid fuzzy GBML algorithm. Our hybrid fuzzy GBML algorithm [8] has a Pittsburgh framework where a rule set is handled as an individual. A Michigan-style algorithm is probabilistically used as a kind of local search after genetic operations in the Pittsburgh-style framework.

The procedure of our hybrid fuzzy GBML algorithm is summarized as following steps [8]:

[Hybrid Fuzzy GBML Algorithm (Pittsburgh Part)]

Step 1: Generate an initial population of N_{pop} rule sets.
Step 2: Evaluate each rule set in the initial population.
Step 3: Generate N_{pop} rule sets by selection, crossover and mutation.
Step 4: Apply the Michigan part to each new rule set with a pre-specified probability.
Step 5: Evaluate the newly generated N_{pop} rule sets.
Step 6: Construct the next population by choosing the best N_{pop} rule sets from the N_{pop} rule sets in the current population and the newly generated N_{pop} rule sets.
Step 7: If a pre-specified termination condition is satisfied, terminate the execution of this algorithm. Otherwise, return to Step 3. In our computational experiments, the total number of generations is used as the termination condition.

[Michigan Part]

Step A: Let a new rule set in Step 4 of the Pittsburgh part be *S*, which is used as the current population in the Michigan part.

Step B: Classify training patterns by *S*. Then calculate the number of correctly classified training patterns by each fuzzy rule, which is used as the fitness value of each fuzzy rule.

Step C: Generate k fuzzy rules where k is an integer satisfying the inequality $5(k\text{-}1) < |S| \leq 5k$.

Step D: Remove the worst k fuzzy rules from *S*. Then add the newly generated k fuzzy rules to *S*.

Step E: Return the updated *S* to the Pittsburgh part where *S* is used as a newly generated rule set.

C. Parallel Distributed Implementation of Hybrid Fuzzy GBML algorithm

In this study, for large data sets, we use parallel distributed implementation of our hybrid fuzzy GBML algorithm [18]. There are two main characteristics of our parallel distributed model. One is that not only a population but also training data are divided into a number of subpopulations and training data subsets, respectively. A pair of a subpopulation and a training data subset is assigned to a single CPU core. Since a number of GBML algorithms are performed in parallel at a workstation with multiple CPU cores, we can drastically reduce the computation time. The other characteristic is the use of training data rotation and individual migration. These two operations avoid the overfitting to the training data and improve the search performance. See [18] in detail.

V. Computational Experiments

The effects of inhomogeneous interval and fuzzy partitions are examined through computational experiments using various benchmark data sets in this section.

A. Experimental Settings

The operators and parameters used in our hybrid fuzzy GBML algorithm and its parallel distributed model are listed as follows:

Population size: 210,
Crossover: uniform crossover,
Crossover probability: 0.9 (Pittsburgh), 0.9 (Michigan),
Mutation probability: $1/(n \times |S|)$ (Pittsburgh),
$1/n$ (Michigan),
n: Number of dimensions,
Michigan operation probability: 0.5,
Initial number of rules: 30,
Max number of rules: 60,
Termination condition: 10,000 generation (non-parallel),
50,000 generation (parallel),
Number of subpopulation: 7,
Number of training data subset: 7,
Rotation interval: 100 generations,
Migration interval: 100 generations.

As we mentioned before, we used inhomogeneous fuzzy partitions derived from interval ones as an antecedent sets in this study. For the setting of fuzzification grade F, we examined six specifications of F, that is, $F = 0.0$, 0.2, 0.4, 0.6, 0.8, and 1.0 to examine the effectiveness of the fuzziness. Here, $F = 0.0$ means interval partitions. On the other hand, $F = 1.0$ means fully-fuzzified partitions. We examined each fuzzification grade for twelve datasets listed in Table I. The data sets used in this paper are available from the KEEL data set repository [19]. We used our hybrid fuzzy GBML algorithm for small data sets from Iris data to Vehicle data. We also used our parallel distributed model for other data sets (i.e., Segment, Phoneme, and Satimage).

TABLE I
Twelve Data Sets Used in This Paper

Data Sets	Patterns	Attributes	Classes
Iris	150	4	3
Wine	178	13	3
Glass	214	9	7
Newthyroid	215	5	3
Heart	270	13	2
Ecoli	336	7	8
Wdbc	569	30	2
Pima	768	8	2
Vehicle	864	18	4
Segment	2310	19	7
Phoneme	5404	5	2
Satimage	6435	36	7

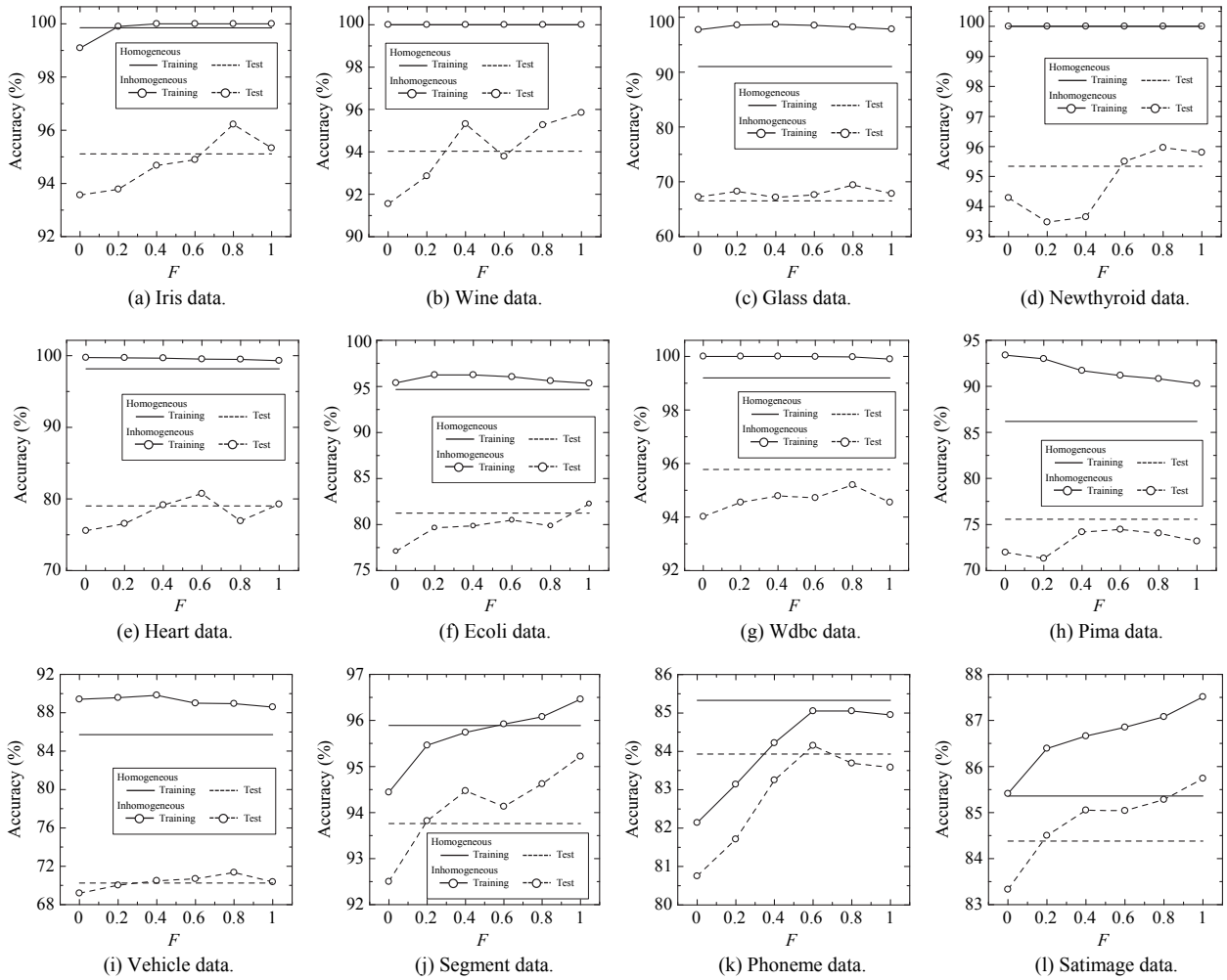
The ten-fold cross-validation was applied to each data set three times using different divisions into ten subsets of the same size. That is, we executed our hybrid fuzzy GBML 30 times for each data set and each fuzzification grade.

B. Experimental Results

In Fig. 5, we show the obtained accuracy for twelve data sets. In each plot, horizontal lines represent the average accuracy by homogeneous fuzzy partitions in Fig. 1. A solid line means the training data accuracy. A dashed line means the test data accuracy. The open circles represent the accuracy by inhomogeneous partitions.

We can see that better training data accuracy was obtained by using inhomogeneous fuzzy and interval partitions than by using homogeneous fuzzy partitions for almost all data sets. Especially, for Glass, Wdbc, Pima, and Vehicle data sets, we can see much improvement by using inhomogeneous partitions. This is because all inhomogeneous partitions were appropriately adjusted by using the training data.

With respect to the test data accuracy, we can observe that the generalization ability was improved by inhomogeneous partitions for ten out of twelve data sets. We can see that better results were obtained with larger fuzzification grade (e.g., $F > 0.5$). This is because the fuzzification of interval partitions leads to the alleviation of the overfitting to the training data.



(a) Iris data. (b) Wine data. (c) Glass data. (d) Newthyroid data.

(e) Heart data. (f) Ecoli data. (g) Wdbc data. (h) Pima data.

(i) Vehicle data. (j) Segment data. (k) Phoneme data. (l) Satimage data.

Fig. 5. The training and test data accuracy for each data set.

In Fig. 6, we show the number of rules in the obtained classifier. In each plot, a horizontal line represents the number of rules in the classifier obtained by homogeneous fuzzy partitions. The open circles represent the number of rules in the classifier obtained by inhomogeneous partitions.

From Fig. 6, we can observe that the number of rules with inhomogeneous partitions was larger than that of rules with homogeneous ones. This may be because inhomogeneous partitions tend to cover well-fitted smaller regions than homogeneous fuzzy partitions. From the comparison between the small fuzzification grade and the large one, we can say the same thing that we need a large number of interval rules to generate an accurate classifier.

From the comparison between the non-parallel model (i.e., (a)-(i) in Fig. 5) and the parallel distributed one (i.e., (j)-(l) in Fig. 5), a different tendency was observed. We need further investigation whether this difference is caused by the size of data sets or the different structure of the algorithms.

VI. Conclusions

In this paper, we examined the effects of inhomogeneous fuzzy partitions derived from interval partitions in our hybrid fuzzy GBML algorithm through computation experiments. We also examined the effect of different fuzzification grades on the obtained classifiers. The experimental results showed the accuracy improvement by using inhomogeneous fuzzy partitions but the increase in the number of rules. Of course, the appropriate fuzzification grade strongly depends on the data sets. Thus we need to carefully specify the grade. One idea is to use repeated double cross validation for parameter choice [20]. This is one of future research topics.

References

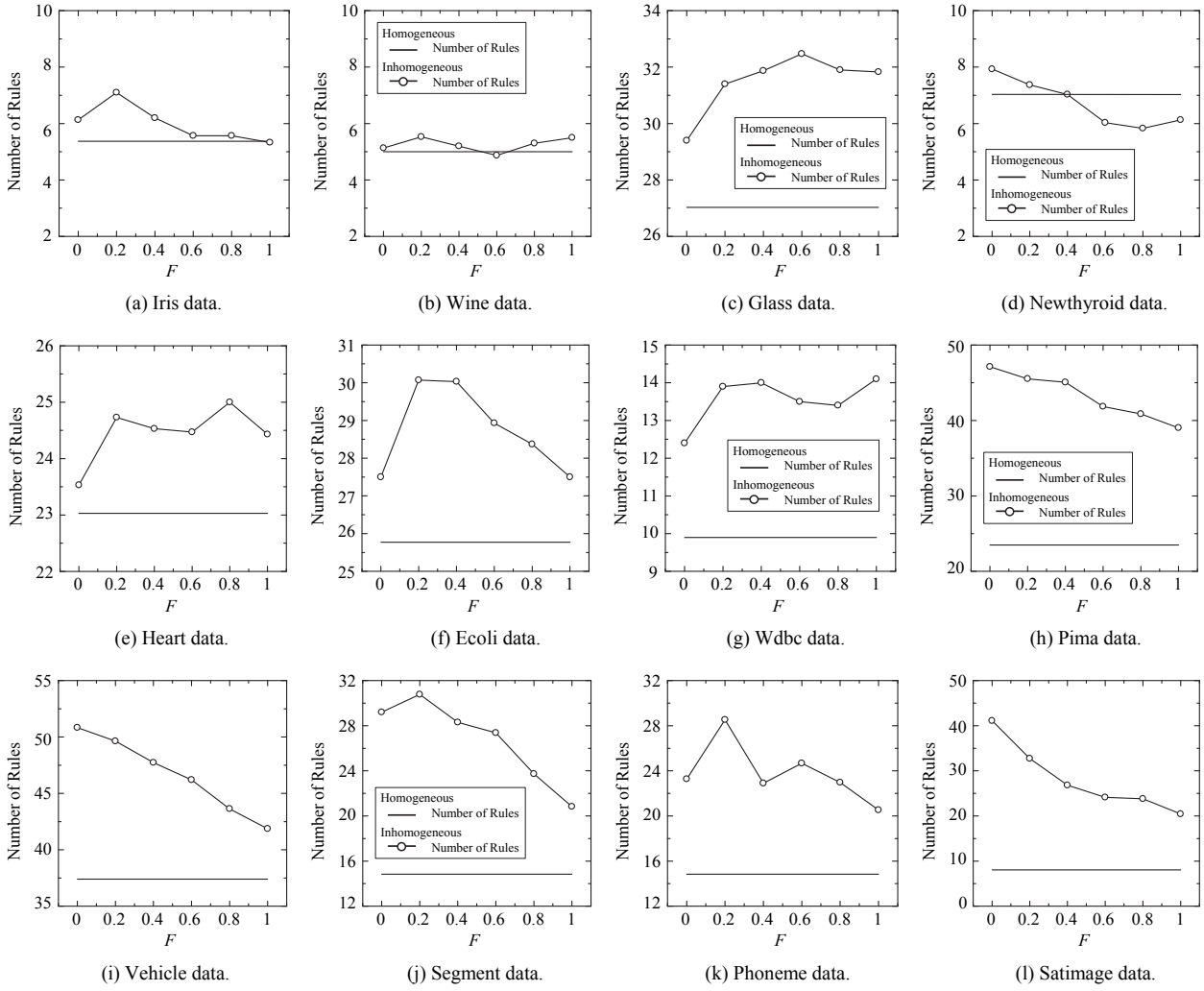
[1] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA, 1989.

[2] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[3] U. M. Fayyad and K. B. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," *Proc. of 13th International Joint Conference on Artificial Intelligence*, pp. 1022-1027, 1993.

[4] J. Dougherty, R. Kohavi, and M. Sahami, "Supervised and unsupervised discretization of continuous features," *Proc. of 12th International Conference on Machine Learning*, pp. 194-202, 1995.

[5] O. Cordon, F. Herrera, F. Gomide, F. Hoffman, "Ten years of genetic fuzzy systems: current framework and new trends," *Fuzzy Sets and*



(a) Iris data. (b) Wine data. (c) Glass data. (d) Newthyroid data.

(e) Heart data. (f) Ecoli data. (g) Wdbc data. (h) Pima data.

(i) Vehicle data. (j) Segment data. (k) Phoneme data. (l) Satimage data.

Fig. 6. Number of rules in the obtained classifier for each data set.

Systems, vol.14, no. 141, pp. 5-31, 2004.

[6] F. Herrera, "Genetic fuzzy systems: Status, critical considerations and future directions," *International Journal of Computational Intelligence Research*, vol. 1, pp. 59-67, 2005.

[7] H. Ishibuchi, T. Nakashima, and T. Murata, "Three-objective genetics-based machine learning for linguistic rule extraction," *Information Sciences*, vol. 136, no. 1-4, pp. 109-133, August 2001.

[8] H. Ishibuchi, T. Yamamoto, and T. Nakashima, "Hybridization of fuzzy GBML approaches for pattern classification problems," *IEEE Trans. on Systems, Man, and Cybernetics - Part B: Cybernetics*, vol. 35, no. 2, pp. 359-365, April 2005.

[9] H. Ishibuchi and T. Yamamoto, "Performance evaluation of fuzzy partitions with different fuzzification grades," *Proc. of 2002 IEEE International Conference on Fuzzy Systems*, pp. 1198-1203, 2002.

[10] H. Ishibuchi, Y. Nojima, "Comparison between Fuzzy and Interval Partitions in Evolutionary Multiobjective Design of Rule-Based Classification Systems," *Proc. of 2005 IEEE International Conference on Fuzzy Systems*, pp.430-435, Reno, USA, May 22-25, 2005.

[11] H. Ishibuchi, K. Nozaki, and H. Tanaka, "Distributed representation of fuzzy rules and its application to pattern classification," *Fuzzy Sets and Systems*, vol. 52, no. 1, pp. 21-32, November 1992.

[12] H. Ishibuchi, T. Nakashima, and M. Nii, *Classification and Modeling with Linguistic Information Granules: Advanced Approaches to Linguistic Data Mining*, Springer, Berlin.

[13] H. Ishibuchi and T. Yamamoto, "Rule weight specification in fuzzy rule-based classification systems," *IEEE Trans. on Fuzzy Systems*, vol. 13, no. 4, pp. 428-435, August 2005.

[14] H. Ishibuchi and T. Nakashima, "Effect of rule weights in fuzzy rule-based classification systems," *IEEE Trans. on Fuzzy Systems*, vol. 13, no. 4, pp. 506-515, 2001.

[15] H. Ishibuchi, T. Nakashima, and T. Morisawa, "Voting in fuzzy rule-based systems for pattern classification problems," *Fuzzy Sets and Systems*, vol. 103, no. 2, pp. 223-238, April 1999.

[16] J. R. Quinlan, "Improved use of continuous attributes in C4.5," *Journal of Artificial Intelligence Research*, vol. 4, pp. 77-90, 1996.

[17] A. Fernández, S. García, J. Luengo, E. Bernadó-Mansilla, and F. Herrera, "Genetics-based machine learning for rule induction: State of the art, taxonomy, and comparative study," *IEEE Trans. on Evolutionary Computation*, vol. 14, no. 6, pp. 913-941, December 2010.

[18] H. Ishibuchi, S. Mihara, and Y. Nojima, "Parallel distributed hybrid fuzzy GBML models with rule set migration and training data rotation," *IEEE Trans. on Fuzzy Systems*, vol. 21, no.2, April 2013.

[19] J. Alcalá-Fdez, L. Sánchez, S. García, M. J. del Jesus, S. Ventura, J. M. Garrell, J. Otero, C. Romero, J. Bacardit, V. M. Rivas, J. C. Fernández, and F. Herrera, "KEEL: A software tool to assess evolutionary algorithms for data mining problems," *Soft Computing*, vol. 13, no. 3, pp. 307-318, February 2009.

[20] H. Ishibuchi and Y. Nojima, "Repeated double cross-validation for choosing a single solution in evolutionary multi-objective fuzzy classifier design," *Knowledge-Based Systems*, vol. 54, pp. 22-31, December 2013.